Attorney Docket No.: 020897-000110US

PATENT APPLICATION

SYSTEM-WIDE OPTIMIZATION INTEGRATION MODEL

Inventor: Carlos M. Collazo, a citizen of The United States, residing at
580 Harbor Colony Court
Redwood Shores, CA 94065

Assignee: MetiLinx
999 Baker Way
Suit 410
San Mateo, CA 94404

Entity: Small business concern

TOWNSEND and TOWNSEND and CREW LLP
Two Embarcadero Center, 8th Floor
San Francisco, California 94111-3834
Tel: 415-576-0200

SYSTEM-WIDE OPTIMIZATION INTEGRATION MODEL

CLAIM OF PRIORITY

[01] This application claims priority from U.S. Provisional Patent Application No. 60/243,783, filed October 26, 2000.

CROSS-REFERENCES TO RELATED APPLICATIONS

[02] This application is related to the following co-pending applications, each of which is incorporated by reference as if set forth in full in this application:

[03] U.S. Patent Application entitled "Multiplatform Optimization Model" (020897-000120US) filed on _____ Serial No. _____ [TBD]; U.S. Patent Application entitled "Aggregate System Resource Analysis and Diagnostics" (020897-000130US) filed on _____ Serial No. _____ [TBD]; U.S. Patent Application entitled "Correlation Matrix-Based on Autonomous Node and Net Analysis Over Disparate Operating Systems" (020897-000140US) filed on _____ Serial No. _____ [TBD]; and U.S. Patent Application entitled "Merit-Based Metric Analysis and Diagnostics of System Resource Model" (020897-000150US) filed on _____ Serial No. _____ [TBD].

BACKGROUND OF THE INVENTION

[04] Digital computer networks, such as the Internet, are now used extensively in many aspects of commerce, education, research and entertainment. Because of the need to handle high volumes of traffic, many Internet sites are designed using several, or many, server computers in a multi-tiered, or "n-tiered," system. An example of an n-tiered system is shown in Fig. 1.

[05] In Fig. 1, n-tiered system 10 includes four major tiers. These are communications tier 12, web tier 14, application tier 16 and database tier 18. Each tier represents an interface between a group of server computers; or other processing, storage or communication systems. Each interface handles communication between two groups of server computers. Note that the tiers are significant in that they represent the communication protocols, routing, traffic control and other features relating to transfer of information between the groups of server computers. As is known in the art, software and hardware is used to perform the communication function represented by each tier.

[06] Server computers are illustrated by boxes such as 20. Database 22 and Internet 24 are represented symbolically and can contain any number of servers, processing systems or other devices. A server in a group typically communicates with one or more computers in adjacent groups as defined and controlled by the tier between the groups. For example, a request for information (e.g., records from a database) is received from the Internet and is directed to server computer 26 in the Web-Com Servers group. The communication takes place in communications tier 12.

[07] Server computer 26 may require processing by multiple computers in the Application Servers group such as computers 20, 28 and 30. Such a request for processing is transferred over web tier 14. Next, the requested computers in the Application Servers group may invoke computers 32, 34, 36 and 38 in the Database Servers group via application tier 16. Finally, the invoked computers make requests of database 22 via database tier 18. The returned records are propagated back through the tiers and servers to Internet 24 to fulfill the request for information.

[08] Of particular concern in today's large and complex n-tiered systems is the performance monitoring and optimization of the system. However, the prior art approaches focus on monitoring and optimization within a single tier. These approaches often require additional hardware, software and database redundancy that is complex, consumes resources, requires time-consuming installation, configuration and operator training. The prior art systems are not easily scalable and have not achieved the desired reliability, performance improvement, security, speed and efficiency.

[09] Thus, it is desirable to provide a system that improves upon one or more of the above-listed (or other) shortcomings in the prior art.

BRIEF SUMMARY OF THE INVENTION

[10] The present invention is a network node health assessment system for multiple networked computers. Each computer, or node, in the network can be equipped with a software process called an intelligence object. The intelligence object reports on characteristics of its host computer system. The object can also report on the data flow to/from the computer system and within the computer system. The intelligence objects can be used on different platforms having any type of hardware (e.g., CPU, peripherals) or software (e.g., operating systems). The core algorithms and logic of the intelligence objects and other software are adaptive and stochastic.

[11] One application of the invention is in a multi-tiered server system. The invention accurately assesses performance across the multiple tiers. The performance information can be used to optimize the server system. The invention provides an efficient user interface for installing, configuring and operating various features. An application programming interface allows users to integrate and adapt the facility for use with any system.

[12] Intelligence objects operate at the server node level to dynamically analyze system processes at each server node. The analysis of system processes is extensive and includes hardware, software, operating system and communications. The result of each intelligence object analysis is communicated to all other intelligence objects dynamically throughout each tier.

[13] A preferred embodiment discloses a method for monitoring the performance of a digital networked system, wherein nodes are executing in components in the networked system, wherein the nodes provide information on at least one aspect of the functioning of a component in the server system, wherein the nodes are organized as multiple groups. The method includes generating a value indicating performance of a first component by a node in a first group; transferring the value to a second node in a second group; modifying the value to indicate performance of a second component; and using the modified value to indicate performance of the digital networked system. In one embodiment the invention provides a method for monitoring the performance of a multi-server system, wherein groups of servers are organized as groups of communication exchange, the method comprising generating a value indicating performance of a first server in a first group; transferring the value to a second server in a second group; modifying the value to indicate performance of the second server; and using the value to indicate performance of the multi-server system.

BRIEF DESCRIPTION OF THE DRAWINGS

[14] Fig. 1 shows a prior art n-tiered system;

[15] Fig. 2A shows intelligence objects and performance value passing in the present invention;

[16] Fig. 2B illustrates architectural components of the present invention.

[17] Fig. 3A illustrates a user interface display to set up node resource pools;

[18] Fig. 3B illustrates a user interface where a user has added specific nodes;

[19] Fig. 3C illustrates the representation of intelligence objects;

[20] Fig. 3D illustrates further organizing of nodes in NRPs into Functional Resource Pools;

[21] Fig. 3E illustrates establishing connectivity and data flow among NRPs, FRPs and nodes;

[22] Fig. 3F illustrates a connection made between FRP 1 and FRP 2;

[23] Fig. 3G shows a subnetwork;

[24] Fig. 3H illustrates a screen shot of a user interface display to allow a user to set-up a DASPO;

[25] Fig. 4A illustrates the Node Listing console;

[26] Fig. 4B illustrates the Graphic View console;

[27] Fig. 4C illustrates the Monitor console;

[28] Fig. 4D illustrates a series graph of the Monitor Console;

[29] Fig. 4E illustrates a balance graph of the Monitor Console;

[30] Fig. 4F illustrates the History Monitor;

[31] Fig. 5A shows the Redirector Deployment and Installation window;

[32] Fig. 5B illustrates the redirector's Remote Set-Up window;

[33] Fig. 5C shows the File Transfer Settings for a file transfer protocol tab;

[34] Fig. 5D shows a destination folder where redirector files are transferred;

[35] Fig. 5E shows a destination folder specified when using a shared network drive to transfer files;

[36] Fig. 5F shows dialog pertaining to launching a remote set-up using a telnet protocol;

[37] Fig. 5G illustrates a portion of the user interface for preparing a redirector;

[38] Fig. 5H shows an HTTP Redirector Configuration screen;

[39] Fig. 5I shows a Create Connection dialog;

[40] Fig. 5J shows a Load Data Link File dialog;

[41] Fig. 5K shows the Data Link Properties window;

[42] Fig. 5L shows the Confirmation dialog;

[43] Fig. 5M shows the Confirmation dialog with security turned on;

[44] Fig. 5N shows the SLO Deployment and Installation window;

[45] Fig. 5O shows the Remote SLO Set-up window;

[46] Fig. 5P is a first illustration specifying controls and parameters for transfer and remote execution functions;

[47] Fig. 5Q is a second illustration specifying controls and parameters for transfer and remote execution functions;

[48] Fig. 5R is a third illustration specifying controls and parameters for transfer and remote execution functions;

[49] Fig. 5S is a fourth illustration specifying controls and parameters for transfer and remote execution functions;

DETAILED DESCRIPTION OF THE INVENTION

[50] A preferred embodiment of the present invention is incorporated into products, documentation and other systems and materials created and distributed by MetiLinx, Inc. as a suite of products referred to as "Metilinx iSystem Enterprise" system. The Metilinx system is designed to optimize digital networks, especially networks of many computer servers in large Internet applications such as technical support centers, web page servers, database access, etc.

[51] The system of the present invention uses software mechanisms called "intelligence objects" (IOs) executing on the various servers, computers, or other processing platforms, in a network. The intelligence objects are used to obtain information on the performance of a process or processes, hardware operation, resource usage, or other factors affecting network performance. Values are passed among the intelligence objects so that a composite value that indicates the performance of a greater portion of the network can be derived.

[52] Fig. 2A illustrates intelligence objects and value passing. In Fig. 2A, intelligence objects such as 102 and 104 reside in computer servers. Any number of intelligence objects can reside in a server computer and any number of server computers in the n-tiered system can be equipped with one or more intelligence objects. A first type of intelligence object (IO) is a software process called a system level object (SLO) that can monitor and report on one or more aspects of other processes or hardware operating in its host computer server. A second type of intelligence object, called a transaction level object (TLO) is designed to monitor transaction load with respect to its host computer or processes executing within the host computer.

[53] In one embodiment, IO 102 measures a performance characteristic of its host computer and represents the characteristic as a binary value. This value is referred to as the "local" utilization value since it is a measure of only the host computer, or of transaction information relating to the host computer. The local utilization value is passed to IO 104. IO

104 can modify the passed value to include a measurement of its own host computer. The modified value is referred to as a "composite" utilization value. The composite utilization value can, in turn, be passed on to other intelligence objects that continue to build on, or add to, the measurements so that performance across multiple computer, tiers, operating systems, applications, etc., is achieved.

[54] Ultimately, the utilization value, or values, is passed on to other processes which can display the result of the combined measurements to a human user, use the result to derive other results, use the result to automate optimization of the n-tiered system, or use the result for other purposes. One aspect of the invention provides for redirecting processes and interconnections on the network based on the assessed utilization values of the computers, or nodes, in order to improve, or optimize, network performance. The processes that perform the redirection are referred to as "process redirection objects" (PROSE).

[55] Note that although the invention is sometimes discussed with respect to a multi-tiered server arrangement that any arrangement of servers, computers, digital processors, etc., is possible. The term "processing device" is used to refer to any hardware capable of performing a function on data. Processing devices include servers, computers, digital processors, storage devices, network devices, input/output devices, etc. Networks need not be in a multi-tiered arrangement of processing devices but can use any arrangement, topology, interconnection, etc. Any type of physical or logical organization of a network is adaptable for use with the present invention.

[56] Fig. 2B illustrates one possible arrangement of more specific components of the present invention. Note that the term "component" as used in this specification includes any type of processing device, hardware or software that may exist or may be executed within or by a digital processor or system.

[57] Systems such as those illustrated in Figs. 1, 2A and 2B, along with virtually any type of networked system, can be provided with IOs. In a preferred embodiment, the IOs are installed on each server in the network in a distributed peer-to-peer architecture. The IOs, along with aggregation software, discussed below, measure real-time behavior of the servers components, resources, etc. to achieve an overall measure of the behavior and performance of the network. A preferred embodiment rates and aggregates network components using a system-wide model discussed in the related applications discussed, above.

[58] The preferred embodiment collects data on low-level system and network parameters such as CPU utilization, network utilization, latency, etc. The data is produced and shared in small four-byte values. In a hierarchy set up by an administrator, or

automatically configured by the system, a value is combined with other values to achieve a composite value. The composite value is then passed along the hierarchy and used to obtain further composited values so that overall system performance is ultimately provided in the composited values.

[59] A network set up with the IOs and other monitoring, analysis and optimization tools as discussed herein is referred to as a Dynamic Aggregate System Process Optimization (DASPO) network. There are three basic phases of operating a DASPO to achieve network improvement or optimization. These phases are (1) set-up, (2) analysis and (3) optimization. In a preferred embodiment, the system of the present invention provides various user tools, including console interfaces, to allow a human user to participate in the different phases. However, provision is also made for automating the different phases to varying degrees.

[60] The operation and implementation of the three phases is heavily dependent on the system-wide model employed by the present invention. The system-wide model is discussed, below, in connection with the three phases and user interfaces for controlling the three phases.

Set-Up

[61] There are five basic steps in setting up a DASPO network, as follows:

Define Node Resource Pools (NRPs)

Add Nodes

Install Intelligence Objects on Selected Nodes

Define Functional Resource Pools (FRPs); and

Establish Connectivity and Data Flow

[62] Fig. 3A illustrates a user interface display to set up node resource pools. In Fig. 3A, node pools are displayed as ovals with labels. NRPs are used to group nodes for organizational purposes. NRPs are used in place of the tier illustration approach of Figs. 1A and 2A. NRPs can be used to create the equivalent of a tiered structure, or they can be used to create other structures of nodes. Fig. 3A shows a Web Server Pool and a Data Server Pool. An Application Server Pool, or other, user defined pool, can be created and labeled. Any number of pools can be defined.

[63] Fig. 3B illustrates a user interface where a user has added specific nodes to the defined NRPs. Nodes can be added by selecting them individually from an existing domain, or by providing specific internet protocol (IP) addresses. A preferred embodiment of the

invention uses nodes that follow standard internet conventions such as machine, or IP, addresses. However, other embodiments may use other protocols, standards, etc., to define nodes. Node names can be generic, as shown in Fig. 3B, or they can be given unique names by a user, or assigned automatically. Naturally, any number and type of node can be assigned to a pool. The pool/node hierarchy is displayed and manipulated much like a familiar file management system.

[64] Fig. 3C illustrates the representation of intelligence objects (IOs). IOs are defined and associated with nodes. Two types of IOs are provided in a preferred embodiment. These are the System Level Object (SLO) and Transaction Level Object (TLO). Each IO is typically identified by the icon to the left of the descriptive text. The icon is placed adjacent to a node in which, or to which, the IO corresponds. During operation, the IO gathers information on the operation and resource use of components at the node.

[65] SLOs can be grouped into pools. The preferred embodiment provides two types of pools as (1) Functional Resource Pools to organize SLOs for nodes that support a common application so that nodes with like functionality are grouped; and (2) Node Resource Pools for organizing FRPs and SLOs for nodes that provide a common service. Links between pools and nodes indicate where functional relationships exist. NRPs and FRPs link together to provide system process flow and to define sub networks for optimization calculations.

[66] Fig. 3D illustrates organizing of nodes in NRPs into Functional Resource Pools.

[67] Once NRPs have been created and nodes assigned, the NRPs can be further subdivided into Funcitonal Resource Pools (FPRs). The FRPs provide a refinement of node function by allowing nodes to be grouped according to specific roles assigned to the FRPs (i.e., Managerial Login servers, Staff Login servers, etc). One or more FRPs can be created inside a NRP, as shown in Fig. 3D. In a preferred embodiment, only SLO and TLO nodes can belong to an FRP.

[68] Fig. 3E illustrates establishing connectivity and data flow among NRPs, FRPs and nodes.

[69] An important step in configuring a network involves determining the route that transactions will take when they move through the system. Routes are determined by the way pools and nodes are linked together. There are three different levels at which links can be defined, as follows:

a. Node Resource Pool to Node Resource Pool

b. Functional Resource Pool to Functional Resource Pool

c. Node to Node

[70] In a DASPO network, NRPs represent the lowest level of detail and nodes represent the highest level. Connections made at higher levels of detail will override the connections made at lower levels. Linking also has certain important implications. For example, if two NRPs are linked, the inference is made that every FRP and every node within the two pools is connected, as shown in Fig. 3E.

[71] Network management is simplified by allowing connections to be made at different levels. Initial connections can be made quickly and simply when establishing an initial network transaction process flow since higher level connections automatically define lower-level connections. For example, a pool-to-pool connection automatically defines lower FRP and node connections with respect to FRPs and nodes within the connected pools. As more network fine-tuning becomes necessary, a refinement of the initial set of links, at a more detailed level, is possible (i.e. node-to-node).

[72] Defining network connections results in the creation of DASPO subnetworks. A DASPO subnetwork is a specific relationship defined between nodes that are linked together across Functional Resource Pools. Subnetworks can, but need not, have a correlation to the physical or logical network organization. For example, subnetworks can follow the multi-tiered design discussed above where each of three subnetworks corresponds to web, application and database tiers. The concept of subnetworking allows a user to flexibly define transaction flows across a network when calculating ideal system optimization.

[73] Fig. 3F illustrates a connection made between FRP 1 and FRP 2. This creates a subnetwork among nodes associated with the FRPs. A subnetwork exists from the "A" node as shown in Fig. 3G. The "A" subnetwork includes nodes B and C from FRP 2.

[74] When nodes are grouped together in Functional Resource Pools, their SLOs and TLOs communicate Local Node Value (LNV) and other intelligence object information to each other. As a result of this communication, each node is aware of the value of every other node in its FRP and, if queried, can identify the Best Node. The Best Node is defined as the server within a particular FRP that is able to handle a system transaction with the greatest efficiency at a given moment. A detailed description of value formats, value passing, composite values and other uses of values can be found in related patent application (3), cited above.

[75] From the LNV of a first node, and from the LNVs of other nodes related to the first node in a subnetwork, a Composite Node Value (CNV) is calculated. A preferred embodiment of the invention uses normalized weights to rank the contribution of the LNV and CNV of every node in the subnetwork associated with the first node. The preferred embodiment takes network latency into account to modify passed CNV and/or LNV values when the values are passed to different nodes.

[76] One feature of a preferred embodiment is that the nodes gather data in the form of CNVs and LNVs and the data is accumulated by a central console, or computer system, operable or accessible to a human user for monitoring and control. This approach allows a administrator to monitor, log, analyze, adjust and optimize separate aspects of a network system. Past, recent and current performance of the network is provided. The network can be automatically instructed by the console (or another system or process) to act in accordance with measured parameters (e.g., based on the CNV and LNV data) to redirect data transfers to the best available resources, nodes, or other components. This approach of distributed, hierarchical, peer-to-peer value gathering to a central console provides efficient and accurate system management.

[77] When DASPO subnetworks are created, an FRP process has information on the best node to utilize at any point in time. The "best node" may not necessarily be the the least utilized node. By providing a global view of system performance, an FRP process can determine nodes which, if routed to, would provide overall system performance improvement. Similarly, an FRP is aware of best nodes for routing or other utilization in the FRP's subnetwork, allowing for faster rerouting decisions and improved resource utilization.

[78] Fig. 3H illustrates a screen shot of a user interface display to allow a user to set-up a DASPO.

[79] In Fig. 3H, the features discussed above are shown, including the use of pools, FRPs and SLOs interconnected to form subnetworks. Area 120 is used to set up subnetworks. Area 122 is used to define interconnections. Area 124 is used to provide details on objects and to allow a user to easily select objects for use.

Analysis

[80] Analysis includes monitoring and administration functions. Users can view results of node data-gathering which indicates the performance of system components, transfers, etc. Various administrative functions can be performed such as saving and modifying configurations, scheduling events,

[81] Four consoles, or basic types of interfaces, are used to help direct network optimization and manage the administration. The consoles are as follows:

1. Node Listing Console
2. Graphic View Console
3. Monitor Console
4. History Monitor Console

[82] Fig. 4A illustrates the Node Listing console.

[83] The Node Listing console provides a list of all the network nodes that are part of the current loaded network configuration, as well as the current status of those nodes. The console is also the location from which user access can be managed; different network configurations can be saved and loaded; backups can be initiated, and Wizards, or automated assistance, for redirectors and System Level Objects (SLOs) can be started.

[84] Fig. 4B illustrates the Graphic View console.

[85] The Graphic View console allows users to visually identify and manipulate the various nodes, pools and connections in a DASPO network in an easy-to-use graphical user interface.

[86] Fig. 4C illustrates the Monitor console. The Monitor console is a real-time tracking feature that measures the available processing capacity of selected nodes in DASPO network to help assess node performance. The node information is displayed in a simple graph or bar format, and the data can be tracked and saved for future reference.

[87] The Monitor console can provide several different graphs for visual presentation of information.

[88] Fig. 4D illustrates a series graph of the Monitor Console.

[89] In the series graph, selected SLO and TLO nodes appear with statistical values from 0 to 100 for each node at a given instant in time. The statistical value reflects the current load capacity of the node. The higher the value, the more processing capability is available to be utilized. A lower value indicates an overworked node that has a low processing capacity.

[90] Host nodes that are selected to be monitored will appear in the Host graph. This graph performs identically to the Series graph.

[91] The Percentage graph measures the statistic values of SLO, TLO and Host nodes together on the same graph. This graph performs similarly to the Series and the Host graphs.

[92] Fig. 4E illustrates a balance graph of the Monitor Console.

[93] In the balance graph, statistical differences between the nodes is shown. Examples of types of differences that can be displayed include average, variance, maximum, minimum etc. These variances are shown visually on one or more bar graphs. A list of available balance variables can be selected and applied by a user. This graph appears beneath the Series and the Host graph in the iSystem Enterprise monitor. Note that the Balance graph does not appear when a Mixed Series is selected.

[94] Before node statistics or balance variables can be displayed in the Monitor graphs, the nodes to be monitored must first be selected. There are two selector fields at the bottom of the Monitor screen shown in Fig. 4E. The left-hand selector field 132 is used for adding SLO, TLO or Host nodes. The right-hand selector field 134 is used to add balance variables. (Note: the balance variable selector is not available when a Mixed Series is selected).

[95] Fig. 4F illustrates the History Monitor.

[96] When network nodes are tracked using the Monitor feature the captured data is stored, for future reference, in a log file. This log file can be accessed and displayed at any time using the History Monitor console. The History Monitor also provides a variety of features that allows saved data to be manipulated, displayed and compared in a variety of different ways. Note: In order to use the History Monitor feature, nodes must first be set up and tracked using the Monitor. For more information, see Monitor Console.

[97] The History Monitor provides several graphs similar to those described, above, for the Monitor Console.

[98] The History Monitor includes a series graph where monitored SLO and TLO nodes appear. This graph displays a statistical value (from 0 to 100) for each selected network node at a given instant in time. This statistical value reflects the load capacity of the node. The higher the value, the more processing capability is available to be utilized. A lower value indicates an overworked node that has a low processing capacity.

[99] Monitored Host nodes will appear in the Host graph of the History Monitor. This graph performs identically to the Series graph.

[100] The Percentage graph of the History Monitor displays the monitored statistic values of SLO, TLO and Host nodes together on the same graph. This graph performs identically to the Series and the Host graphs.

[101] The statistical differences between the nodes (i.e. average, variance, maximum, minimum etc.) can be measured in the balance graph of the History Monitor. A list of available balance variables can be selected and applied by a user. This graph appears

beneath the Series and the Host graph in the iSystem Enterprise monitor. Note that the Balance graph does not appear when a Mixed Series is selected.

[102] Before the node statistics that have been captured in the monitor can be displayed in the History Monitor graphs, the nodes to be monitored must first be selected. There are two selector fields at the bottom of the History Monitor screen of Fig. 4F. The left-hand selector field 136 is used for adding SLO, TLO or Host nodes. The right-hand selector field 138 is used to add balance variables. (Note: the balance variable selector is not available when a Mixed Series is selected).

Optimization

[103] Part of the optimization process is accomplished by redirecting requests and connections within Functional Resource Pools. This is achieved using data generated by SLO-nodes, which compute their own statistics and broadcast the results through the pools.

[104] This way of implementing redirection is available to every application implemented in-house. However, there are many pre-packaged applications and objects commonly used, whose code cannot--and probably shouldn't--be altered. These types of applications include web servers and COM-objects. Due to the different nature of requests and connections that take place in a complex network system, specific objects must handle redirection inside each class of calls. A preferred embodiment of the present invention includes objects for redirecting HTTP-requests and OLE DB-connections. However, other embodiments can employ other objects in other environments and on other platforms such as HTTP in Java, DB in C++, etc., on Linux, Solaris, etc.

[105] An HTTP Redirector is a Windows-based application (HTTPRedir.EXE) capable of receiving HTTP-requests and redirecting them to a selected web server according to some predefined selection criteria. Starting from a list of web servers and a selection method, this application gathers load-statistics and availability from the web servers and effectively redirects the requests transparently to the requesting client.

[106] The HTTP Redirector can be used in different ways to accomplish its tasks. Its interaction with clients and web servers depends on the place it's located, the port it's using for listening and the links defined on the accessed pages at the web servers. Issues regarding server affinity, client sessions, etc, must be handled by the web administrator.

[107] OLE DB-Connection Redirector is a DCOM server packed into a Windows-based executable (OLEDBRedir.EXE). This object is able to keep track of the load-statistic of a set of database servers and to supply a predefined connection string corresponding to the

selected database server when requested. This redirector object needs to be alive to monitor the database severs. Therefore, it's necessary that the application be manually started once it's installed. This represents a difference to commonly used automation servers that are automatically activated upon client requests.

[108] The redirector deployment and installation process consists of five main stages:

1. Select nodes for redirector installation
2. Specify server general settings for each node
3. Specify file-transfer and remote-execution settings for each node
4. Execute redirector installation procedure
5. Configure the installed redirector

[109] The remote installation mechanism is built around a Windows application (RSLOSetup.EXE) and a set of auxiliary files that are actually moved to the target node to perform the installation. From this point another mechanism launches the installation process on the remote node. For UNIX/Linux platforms, SLO will be installed as a daemon. For Windows-based platforms, SLO will be installed as a regular application included in the Startup folder for every user.

1. Selecting nodes for redirector installation

[110] Fig. 5A shows the Redirector Deployment and Installation window.

[111] By choosing the control" Select Functional Resource Pool" a list of available FRPs appears from the drop-down menu. "Add Redirector" allows the selection of the IP address for a node that is to be designated as a redirector. "Modify Redirector" allows an existing node to be reconfigured so that a different node takes its place as a redirector, or a different type of redirector (HTTP or DB) is used. "Remove Redirector" removes a server that is highlighted by the user from the Deployment and Installation window.

[112] "Change configuration" allows the installed redirector to be configured for use once nodes have been selected as redirectors and the file transfer and execution is complete. "Install All the Redirectors" is selected after nodes have been chosen for the installation of redirectors. The Install operation takes the user to the Redirectors Remote Setup window where the transfer and execution of redirector files can commence.

2. Specifying Server General Settings

[113] Once nodes have been selected for redirector installation, the Redirectors Remote Setup window opens.

[114] Fig. 5B illustrates the Redirectors Remote Setup window.

[115] The Redirectors Remote Setup window is used to define the operating system, file-transfer and remote-execution mechanisms for each node. (Nodes are referred to as Remote Servers in this window.) Selecting different file-transfer and remote-execution mechanisms will activate corresponding tabs which will appear behind a General Settings tab, discussed below. These new tabs can require separate configuration, as discussed in detail in the next section. Changes to general settings are reflected in the list of nodes in the left-hand Remote Server field.

[116] Note that certain restrictions apply during this portion the setup. For example, DCOM is only available to Windows platforms. In some cases, selecting the option "None" for an operation mechanism is useful. For example, if the corresponding files are already placed on a node (due to a previous attempt to install or because common drives are used), only remote execution is required.

3. Specifying File-Transfer and Remote-Execution Settings

[117] Depending on the file-transfer and remote-execution mechanisms that were selected in previous steps, one or more new tabs appears behind a General Settings tab. Each tab can be "active" and brought to the forefront by clicking on the tab. Fig. 5C shows the File Transfer Settings for file-transfer protocol (FTP) tab. FTP settings require specifying the FTP username and password (if applicable) and the FTP destination directory. By default an anonymous username and the Home directory are set.

[118] When using SLO, the destination folder where the redirector files will be transferred is required, as shown in Fig. 5D. By default, the files will be transferred to the default remote SLO folder.

[119] When using a shared network drive to transfer files, a Destination Folder must be specified, as shown in Fig. 5E. This folder points to a drive (local to the target node) that is shared along the network and mapped locally (at a central point). Common functionalities, such as mapping a network drive or creating a new folder are included. Note that file-transfer operations are carried out using the current user credentials, which means the current user must have enough rights to perform the operations.

[120] When launching a remote setup using the telnet protocol, as shown in Fig. 5F, username and password are required. The Remote Execution Folder points to a local folder (on the remote server) where the setup files were moved during the file-transfer step.

[121] Redirector configuration is the final step in preparing a redirector for use in a DASPO network. Fig. 5G illustrates a portion of the user interface for preparing a redirector.

[122] A Redirector Listening Port is a port number used by the redirector to listen for HTTP requests. Port 80 is used by web servers to listen and by web browsers to connect. It is recommended that this port number be used for the redirector if the redirector will be performing as a web server. It is important to note that only one application can be listening on one port, therefore the redirector cannot coexist with a web server on the same computer if both are listening through the same port. The Check It! button verifies that the selected port number is available, meaning no other local application is currently listening on this port. When configuring the redirector from iSystem Enterprise, the Check it! button is disabled.

[123] A Functional Resource Pool is the source list of web servers. The SLO Address field refers to an SLO-node installed in one of the computers belonging to the pool. Statistics will be retrieved from a single SLO instead of asking individually. To retrieve the list of servers from the SLO-node the Get Servers button is pressed.

[124] The Server Selection Method directs how servers are selected for redirection. Choices include a web server with Best Statistics or in a Round Robin fashion. Note that a server is not be selected if it doesn't contain the requested object, even if its turn has come up for redirection.

[125] A list of web servers available for redirection is displayed. These are the web servers that might receive transaction requests. Web servers can be added, removed or modified using the displayed list. The Remove Selected button removes a selected web server from the list. The removed server is not be included in any further redirection. The Clear Address List button clears all web servers from the list. The Add Server button adds a new web server to the list. The Modify Server button modifies the parameters corresponding to a server in the list.

[126] A preferred embodiment uses a DCOM server packed into a Windows-based executable process called an "OLE DB-Connection Redirector." This object is able to keep track of the load-statistic of a set of database servers and to supply a predefined connection string corresponding to the selected database server when requested. This redirector object must be active to monitor the database severs. Therefore, the application must be manually

started once installed. This is different from commonly used automation servers that are automatically activated upon client requests.

[127] Instead of directly assigning connection strings to their connection objects, developers create a remote instance of the redirector and request a valid connection string from it. Using this connection string guarantees that the best available database server is selected.

[128] The HTTP Redirector Configuration screen is shown in Fig. 5H.

[129] The Functional Resource Pool area is the source list of data base servers. The SLO Address field refers to an SLO-node installed in one of the computers belonging to the pool. Statistics are retrieved from a single SLO instead of asking individually. To retrieve the list of servers from the SLO-node the Get Servers button is pressed.

[130] The Server Selection Method area indicates how servers are selected for redirection. Choices include a database server with the Best Statistic or Round Robin fashion. The Database Connection List displays a list of database servers and connection strings included for redirection. These are the database servers that might receive the redirector connection requests. Items in the list can be added, removed, or modified.

[131] The Remove Selected button removes the selected database connection from the list. The removed connection is not included in any further redirection. The Remove All button is used to remove all connections from the list. The Add DB Connection button adds a database connection to the list. The Modify DB Connection is used to modify the parameters corresponding to a connection in the list.

[132] Once all modifications are introduced, a configuration can be updated by pressing the OK button. Canceling the operation doesn't modify the current configuration.

[133] After clicking on the Add DB Connection button, the Create Connection dialog is shown in Fig. 5I. This dialog allows a new OLE DB connection to a database server to be defined. Connection parameters include a connection string and the name of the server.

[134] The connection string can be typed directly, loaded from a Universal Data Link (UDL) file or edited using the corresponding system dialog. Connection strings can be manually or automatically tested before saving to the current configuration. Automatic testing is performed when the "Test database connection before save" box is checked. The testing process attempts to open a database connection using the given connection string.

[135] Note that there are situations when testing a connection doesn't make sense. This occurs when the redirector and the database server are located on different domains.

Applications requesting a connection might use aliases to reach the database servers and these aliases can be unknown to the redirector.

[136] If the connection string is loaded from a file, then the file is selected using the Load Data Link File dialog, shown in Fig. 5J. This is a common dialog oriented to search for UDL files.

[137] Another possibility is to select the Edit Connection String button, which opens the Data Link Properties window shown in Fig. 5K. This dialog contains a wizard that allows a step-by-step definition of the properties.

[138] After loading from a file or defining through the Data Link wizard, the resulting connection string is loaded into a confirmation dialog, shown in Fig. 5L, which identifies the name of the provider, the parameters and the settings for security. Fig. 5L shows a confirmation dialog when security is turned off. The identification confirms the settings made previously. To change the provider or the parameters, the Modify Parameters button is pressed to return to the system wizard. Security settings can be modified directly in this dialog by selecting different security settings and/or modifying the usemame and password associated to the connection.

[139] Fig. 5M shows the confirmation dialog with security turned on.

[140] In Fig. 5M, once the OK button is pressed, control is returned to the Create Connection dialog, containing the resulting definitions.

[141] The process of modifying an existing database connection includes some of the same steps discussed previously. To launch the process, a connection at the Configuration Dialog is selected and then the Modify DB Connection button is pressed.

System Level Objects

[142] Before system optimization is determined, the value of each node is measured. In order to collect these measurements, intelligence objects (IOs) are deployed across a DASPO network. These intelligence objects gather statistics on the processes and system loads that are generated at each server node. The format, formation and use of the values, statistics and node information is discussed in detail in the co-pending patent applications referenced, above. Node information includes CPU usage, size and usage statistics of memory and storage space, bytes read/written per second, number of threads, number of processes executing at the node, processor queue length, local response time and network response time. Note that many other types of information about the node, node environment, node host, processor, etc., can be included. Also, not all of the listed node information need

be used in order to practice the present invention. In general, any type of information about resource use, performance or other characteristics can be used.

[143] As mentioned, a preferred embodiment of the invention uses two types of intelligence objects called System Level Object (SLOs) and Transactional Level Objects (TLOs). In a preferred embodiment, SLOs are the most commonly deployed intelligence object. Both SLOs and TLOs perform similar information gathering duties, but TLOs have the additional responsibility of providing statistics for any servers where special hosts (i.e., programs that provide data access and security between an application and a database) are set up. Note that a "host" or "host computer" can be any digital processing hardware device, or software process, that can perform a function on data in a network.

[144] Before system optimization can be determined, the value of each node must first be measured. In order to collect these measurements, intelligence objects (IOs) are deployed across a DASPO network. These intelligence objects gather statistics on the processes and system loads that are generated at each server node. The most commonly deployed IO is the System Level Object (SLO).

[145] SLOs can be installed on remote computers from a central point and is able to work across MS-Windows and TCP/IP networks. Installations can be made on computers running Windows 95/98, Windows NT, Windows 2000, Linux and Solaris UNIX. Depending on the platform, configuration and available services on the target machine, installations take place by means of ftp, telnet, network shared drives and/or DCOM.

[146] The installation process consists of four main stages as follows: (1) Selecting target nodes; (2) Specifying server general settings (3) Specifying file-transfer and remote-execution settings for each node and (4) Executing the installation procedure.

[147] The remote installation mechanism is built around a Windows application and a set of auxiliary files that are actually moved to the target computers to perform the installation. The remote installation mechanism consists of two parts--one for transferring files to the server, and another to launch the installation process on the remote server. For UNIX/Linux platforms, SLO is installed as a daemon. For Windows-based platforms, SLO is installed as a regular application included in the Startup folder for every user.

[148] Fig. 5N shows the SLO Deployment and Installation window.

[149] In the Deployment and Installation window, all available network nodes are displayed in the left-hand Computer column. Nodes that are scheduled to have SLO installed will appear in the right-hand computer column.

[150] Select All allows the quick selection of all the nodes in the left-hand Computer column. Invert Selection is used when a long list of nodes is to be added for SLO installation. It is often easier to select the nodes in the left-hand Computer column that that aren't wanted and then press the Invert Selection button. Any selections that have been made will then be inverted. In other words, checked boxes will become unchecked and vice-versa.

[151] Deselect All removes all checkmarks from the nodes selected in the left-hand Computer column. The Add button, adds nodes that have been selected in the left-hand Computer column and adds them to the SLO installation list. Nodes in the right-hand window that have been selected for SLO installation in the network can be removed by being selected and then clicking on the Remove button. Once the desired nodes are selected, the Install button is pressed to start the SLO deployment process.

[152] Once nodes have been selected for SLO installation, the Remote SLO Setup window, shown in Fig. 5O, opens to allow specification of server general settings.

[153] Specification of server general settings defines the operating system, file-transfer and remote-execution mechanisms for each node. (Note: nodes are referred to as Remote Servers in this window.) Selecting different file-transfer and remote-execution mechanisms activates corresponding tabs which appear behind the General Settings tab. These new tabs can require separate configuration. Any changes that are made in the General Settings tab are reflected in the list of nodes in the left-hand Remote Server field.

[154] In the preferred embodiment, restrictions apply during this portion of the SLO setup. For example, DCOM is only available to Windows platforms. In some cases, selecting None for an operation mechanism can make sense. For example, if the corresponding files are already placed on a node (due to a previous attempt to install or because common drives are used), only remote execution is required.

[155] Figs. 5P-S illustrate specifying controls and parameters for file transfer and remote execution functions.

[156] Depending on the file-transfer and remote-execution mechanisms that were selected in previous steps, one or more new tabs will appear behind the General Settings tab. The File-Transfer Settings for FTP tab allow specification of the FTP username and password (if applicable) and the FTP destination directory. By default the Anonymous username and the Home directory are set. The File-Transfer Settings for Shared Network Drives allows a Destination Folder to be selected, for example, when using a shared network drive to transfer files. This folder points to a drive (which is local to the node where SLO will be installed) that is shared along the network and mapped locally (at a central point). Common

functionalities, such as mapping a network drive or creating a new folder are included. Note that file-transfer operations are carried out using the current user credentials, which means the current user must have enough rights to perform the operations.

[157] When launching a remote setup using the telnet protocol, a username and password are required. The Remote Execution Folder points to a local folder (on the remote server) where the setup files were moved during the file-transfer step. The final way to launch SLO setup is using DCOM. During the file-transfer step, all necessary files were sent to a local folder on the remote server. The complete path for this folder should be typed into the Local path in remote computer" field. DCOM allows remote processes to be executed using different user credentials. This parameter is selected in the DCOM User field.

[158] For a successful execution of the remote setup, the selected user must have rights to launch applications and access disk services through DCOM on the remote server. In terms of DCOM security, this means the user (or the group the user belongs to) must be listed in the "Default Access Permissions" (with Allow Access permission) and "Default Launch Permissions" (with "Allow Launch" permission). These lists can be seen and modified by executing the configuration application for DCOM and selecting the "Default Security" tab. For more information consult your DCOM documentation.

[159] Once the parameters are defined for each server, the installation process can begin. To start the installation, the user selects a predetermined icon or button on the user interface. Once the installation process is launched, SLO files are transferred and launched for each specified node. Results, errors and notifications can be viewed under the Results tab as the installation is in progress.

[160] Although the present invention has been discussed with respect to specific embodiments, these embodiments are merely illustrative, and not restrictive, of the invention. For example, although the invention is discussed primarily with reference to multi-tiered, or n-tiered, systems; it should be apparent that aspects of the invention can be used with any type of processing system even where the architecture does not include multiple tiers. Aspects of the invention can also be applied to stand-alone systems, or systems that are not considered networks.

[161] Thus, the scope of the invention is to be determined solely by the appended claims.